IoTeX

A Decentralized Network for Internet of Things Powered by a Privacy-Centric Blockchain

The IoTeX Team (support@iotex.io)

Last Updated: July 12, 2018
Version 1.5

Disclaimer This paper is intended to be a technical overview. It is not intended to be comprehensive nor to be the final design; thus non-core aspects are not covered, such as APIs, bindings or programming languages.

Abstract

The majority of Internet of Things (IoT) devices are deployed in a centralized way as of today, though decentralized in nature. Many problems have been exposed: scalability, high operating cost, privacy concerns, security risks, and lack of functional values. Blockchain, decentralization by design, could be a good solution to these problems. First, blockchain is elastic enough to solve the scalability challenge of IoT in a cost-effective manner. Second, retaining data within well-scoped blockchains eliminates the concern of IoT data being stored in cloud and potentially being leaked or abused. Third, blockchain with smart contract and tokens has huge potential to enable autonomous coordination of devices to create functional values. However, existing blockchains have their limitations facing IoT problems because of the special characteristics of IoT, such as, large quantity and heterogeneity of devices, and constrains in computation, storage and power, etc.

This paper introduces IoTeX, a decentralized network for IoT powered by a privacy-centric blockchain with four major innovations:

- Blockchains in blockchain for a well-balanced distributed network that maximizes scalability and privacy in a cost-effective way;
- True privacy on blockchain based on relayable payment code, constant-size ring signature without trusted setup, and first implementation of bulletproof;
- Fast consensus with instant finality greatly improving the throughput of the network and reducing transactional cost;
- Flexible and lightweight IoTeX-based system architectures purpose-built for key IoT applications across multiple industry sectors.

Contents

List of Figures

List of Tables

1 Internet of Things

The Internet of Things (IoT) is rapidly emerging as the manifestation of the networked society vision – everything that benefits from a connection is connected. Yet, this far-reaching transformation is just the beginning. The number of connected IoT devices is expected to grow by 21% annually, rising to 18 billion in 2022 [10] and the global market of IoT is expected to grow from 170 billion USD in 2017 to 560 billion USD by 2022 [15], at a compound annual growth rate of 26.9%. Though many industry experts and excited consumers have pegged IoT as the next industrial revolution or the next internet, there are three main problems that are holding back the massively development and adoption of IoT.

1.1 Scalability Problem

The majority of IoT devices are connected and controlled in a centralized way as of today. IoT devices are connected to back-end infrastructures on public cloud services or on premise server farms to transmit data and receive control commands. Currently, the scale of IoT is bottlenecked by the scalability and elasticity of these back-end infrastructures, servers and data centers. The substantially high operating cost of running the scale of IoT is unlikely to be covered by the profit from selling devices. Consequentially, many IoT vendors cannot provide cost-effective devices and applications that are scalable and reliable enough for real-world scenarios [35].

1.2 Lack of Privacy

IoT is expected to enable mass participation of end users on mission critical services such as energy, mobility, legal and democratic stability. Privacy challenges originate from the fact that IoT interacts with the physical world in direct and automatic ways, and the amount of data collected will increase substantially when it scales up. Some of the common privacy threats, as enumerated in [37], are:

1. Identification: Associate a (persistent) identifier, *e.g.*, a name and address or a pseudonym of any kind, with an individual;

2. Localization and tracking: Obtain an individual's location through different means;

3. Profiling: Compile information dossiers about individuals to infer interests by association with other profiles and data sources;

4. Privacy-violating interaction and presentation: Conveying private information through a public medium and in the process disclosing it to an unwanted audience;

5. Life cycle transitions: Devices often store massive amounts of data about their own history throughout their entire life cycle that could be leaked during changes of control spheres in a device's life cycle;

6. Inventory attack: The unauthorized collection of information about the existence and characteristics of personal things, *e.g.*, Burglars can use inventory data to check the property to find a safe time to break in;

7. Linkage: Linking different previously separated systems such that the combination of data sources reveals (truthful or erroneous) information that the subject did not disclose to the previously isolated sources and, most importantly, also did not want to reveal.

All these common privacy threats are due to data leak at device level; or, data leak during communication; or, more often, data leak by centralized parties.

1.3 Lack of Functional Values

Most existing IoT solutions lack meaningful value creation. "Being connected" is the most used value proposition. However, simply enabling connectivity does not make a device smart or useful. A greater portion of the value that IoT produces comes from interaction, cooperation, and eventually autonomous coordination of heterogeneous entities. A few good analogies are that individual cells cooperate to build multi-cellular organisms, insects build societies, humans build cities and states. By cooperating, all these individuals unite to build something that has greater value than their own. Unfortunately, according to [29], 85% of legacy devices lack ability to interact or cooperate with each other due to compatibility issues. The data sharing for business and operational insights is nearly impossible.

2 Blockchain

Blockchain technology was introduced in 2008 and its first implementation, *i.e.* Bitcoin, was introduced a year later, in 2009, published in the paper *Bitcoin: A Peer-to-Peer Electronic Cash System* [21] by Satoshi Nakamoto (alias). Essentially, blockchain is a distributed, transactional database that is shared across all the nodes participating in the network. This is the main technical innovation of Bitcoin and it acts as a public ledger for the transactions. Every node in the system has a full copy of

the current chain state, which contains every transaction ever executed. Every block contains a hash of the previous block, linking these two together. The linked blocks become a blockchain.

2.1 Ingredients

A blockchain can be perceived as a four-dimensional continuum that has three horizontal layers including transaction and blocks, consensus, compute interface, and governance, one vertical layer.

Transaction and Blocks

As the lowest horizontal layer, signed transactions are gossiped among all nodes and blocks are generated by full nodes. This is the foundation of blockchain where transferring of digital assets (thus the inherent values) and account security are achieved via crypto primitives like elliptic curve signature, hash function and Merkle tree.

Consensus

The middle horizontal layer manifests the peer-to-peer nature of the blockchain, where all nodes within the network reach consensus on all internal states on chain via techniques like Proof of Work (PoW), Proof of Stake (PoS) and their variants, Byzantine-fault tolerance (BFT) and its variants *etc.* The consensus layer affects scalability the most. PoW is usually considered less scalable as compared to PoS. In addition, this layer heavily impacts security in terms of double spending and other attacks focused on mutating the blockchain states in an unanticipated way.

Compute Interface

The first two horizontal layers form the shape of a blockchain while the Compute Interface layer is critical to make a blockchain useful, which encompasses extensibility and usability. For instances, smart contract has been implemented by Ethereum to enable programmability where one could count on the distributed "world computer" for executing the terms of a contract. Sidechain, together with merged mining, has also been developed intensively to support programmability. Second-layer protocols like Raiden network [25], state channel has been developed to extend the scalability of a blockchain at this layer. In addition, tools, SDKs, frameworks, and GUIs are also extremely important to usability. The Compute Interface layer gives developers the capability to develop decentralized apps (DApps), an essential part of making the blockchain useful and valuable.

Governance

As with organisms, the most successful blockchains will be those that can best adapt to their environments. Assuming these systems need to evolve to survive, initial design is important, but over a long enough timeline, the mechanisms for change are most important, which is known as the vertical layer governance. There are two critical components of governance:

- Incentive: Each group in the system has their own incentives. Those incentives are not always 100% aligned with all other groups in the system. Groups will propose changes over time which are advantageous for them. Organisms are biased towards their own survival. This commonly manifests in changes to the reward structure, monetary policy, or balances of power.

- Coordination: Since it is unlikely all groups have 100% incentive alignment at all times, the ability for each group to coordinate around their common incentives is critical for them to affect change. If one group can coordinate better than another, it creates power imbalances in their favor. In practice, a deciding factor is how much coordination can be done on-chain (*e.g.*, votes to the rules of the system like Tezos [34], or even roll back the ledger if majority stakeholders don't like the change) *vs.* off-chain (such as Bitcoin Improvement Proposals (BIPs) [3]).

2.2 Operational Models

Blockchains can be categorized as permissionless and permissioned depending on how it is operated. For example, Bitcoin is permissionless meaning that anybody can create an address and begin interacting with the network, which is "build trust from trustless". In contrast, the permissioned blockchain is a closed and monitored ecosystem where the access of each participant is defined and differentiated based on role, which is "build trust from less trusted".

There are benefits and drawbacks to each approach. Regardless, all these considerations boil down to fundamental design trade-offs among trust, scalability, computation and complexity. For example, Bitcoin and Ethereum are blockchains built on top of trustless nodes because scalability is strongly desired. Hence, either lots of computation is needed (in the case of PoW) or more sophisticated consensus mechanism is needed. In contrast, Fabric [14] is a permissioned blockchain where all nodes are considered as trusted and have cryptographic identities, *e.g.*, issued by member services like Public Key Infrastructure (PKI), which makes them highly scalable with low computation and a relatively straightforward consensus mechanism.

Table 1: IoT Benefits From Blockchain Properties

Blockchain Property	IoT Benefits From
Decentralization	Scalability, Privacy
Byzantine fault tolerance	Availability, Security
Transparency & Immutability	Anchoring Trust
Programmability	Extensibility

3 Benefits and Challenges of Blockchain and IoT

Sensing and perception, transformation and transmission, and processing are the essence of most intelligent things on this planet. For IoT, while the sensing and perception layer is spontaneously distributed, the latter two are not for the time being, which is the root for most scalability, privacy and extensibility problems. We envision blockchain technology, if it serves as the spinal cord and nervous system of IoT, as the best candidate to address the aforementioned IoT-specific problems.

3.1 Benefits

By embracing blockchain technology, IoT immediately benefits from the following aspects thanks to blockchain's properties including decentralization, Byzantine fault tolerance, transparency and immutability. Table 1 summarizes how IoT benefits from blockchain properties.

Decentralization

Decentralization frees users and devices from centralized controlled and consistent monitoring, thus partially addressing the privacy concern imposed by centralized parties who monopolize the market and try to understand every aspect of user/device for their own benefits, *e.g.*, advertising. Decentralization, under the context of cryptoeconomy, also indicates "elasticity" that is often defined as "the degree to which a system is able to adapt to workload changes by provisioning and de-provisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible". A blockchain and the underlying cryptoeconomy can be designed in a way that is elastic enough and cost-effective enough for IoT scenarios and applications. For example, more blockchain nodes could be spun up if the network has enough computation tasks with enough incentives to perform.

Byzantine Fault Tolerance (BFT)

The objective of Byzantine fault tolerance is to defend against failures in which components of a system fail in arbitrary ways, *i.e.*, not just by stopping or crashing but by processing requests incorrectly, corrupting their local state, and/or producing incorrect or inconsistent outputs. The Byzantine failure models real-world environments in which computers and networks may behave in unexpected ways due to hardware failures, network congestion and disconnection, as well as malicious attacks. BFT property can be leveraged to achieve many desired security properties in the context of IoT, *e.g.*, eliminate man-in-the-middle (MITM) attacks as there is no single thread of communication that can be intercepted and tampered, make Denial of Service (Dos) attacks almost impossible.

Transparency and Immutability

Blockchain provides cryptographic assurances that the data anchored on the chain is always transparency and immutable, which can be useful in many scenarios, *e.g.*, anchor states of the IoT world on the blockchain for the purpose of auditing, notarization and forensic analysis, identity management, authentication and authorization.

Programmability

Bitcoin came with basic programmability to allow a transaction to succeed only if the underlying small script execute successfully. Ethereum enhances this feature to achieve the Turing-complete smart contract which is written in high-level programming language and executed in a small virtual machine known as EVM. This programmability can be and should be extended to IoT devices, some of which currently only have simple and hard-coded logic that can't be further programmed once shipped.

3.2 Challenges

Benefiting from common properties provided by blockchains does not mean every blockchain is suitable for IoT use. In fact, it doesn't seem like any existing public blockchain can be applied to IoT since there are quite a few challenging problems.

Native Privacy Guarantee Is Not Enough

Native privacy guarantees from blockchain can only help address the privacy pain point in IoT to the degree that it retains data on the chain rather than centralized

servers, using pseudonymity. However, if a device's pseudonym is ever linked to its identity, everything it ever did under that pseudonym will now be linked to it.

No Silver Bullet Blockchain Exists

As mentioned above, IoT is a universe of heterogeneous systems and devices with different purposes and capabilities. It is impossible to find a silver bullet blockchain solution that suits most scenarios. For instance, a blockchain for coordinating millions of industrial IoT nodes should focus on high scalability and transaction throughput, while a blockchain for coordinating smart devices at home should focus on privacy and extensibility. At a macro level, the IoT devices as one specie is definitely evolving at a fast pace, *i.e.*, new technologies are integrated, new standards are developed, new devices are manufactured with new capabilities. In contrast, at a micro level, the individual IoT device's capability, purpose and operational environment also keep changing over time.

Chain Operations Are Heavyweight

In the IoT world, many devices are considered as weak nodes because they are:

- Incapable of performing PoW-based mining due to the power and computation constraints;
- Not able to store large amount of data (*e.g.*, gigabyte level, not mentioning terabyte-level and petabyte-level) due to the power and storage constraints;
- Not able to verify all transactions by processing the whole blockchain;
- Not able to connect to peers all the time, depending on its uptime and connectivity quality.

Therefore, most existing blockchains are too heavyweight for IoT.

3.3 Related Work

IOTA, which recently launched, is built based upon unconventional technology known as the tangle [24]. IOTA attempts to decouple the state transition mechanism with the consensus canonicalisation mechanism by throwing away concepts like blocks and chain. Instead, transaction issuers are also transaction approvers and transaction verification is constructed using a Directed Acyclic Graph (DAG) to make transaction fast and zero cost. The efficiency is obtained by losing globally definite states, which makes desired features like Simple Payment Verification (SPV) for light clients and

smart contracts quite challenging. IoT Chain (ITC) [16], another IoT blockchain project based in China, inherits the same tangle structure from IOTA, and thus has the same pros and cons. HDAC [13] is another recently proposed blockchain for IoT in Korea, which partnered with Hyundai Group, will focus on more specific fields in IoT such as device authentication and Machine-to-Machine (M2M) transaction.

4 IoTeX: Design and Architecture Overview

4.1 Design Principle

IoTeX aims to become the privacy-centric and scalable spinal cord and nervous system for IoT. To achieve this and to address the aforementioned challenges, our architecture design has the following principles.

Separation of Duties

Directly connecting all IoT nodes into one single blockchain is a dream that can't become true. Besides the fact that different IoT applications require fundamentally different feature sets of a blockchain, hosting every IoT node on one blockchain makes it grow fast in size and computation, and eventually become too heavyweight for many IoT devices. Instead, a separation of duties makes sure each blockchain interacts with a specific group of IoT nodes, and, at the same time, interacts with other blockchains when needed. This is analogous to the internet – heterogeneous devices first form an intra-connected group, intranet. Smaller intranets can further form a larger intranet, which eventually connects to the backbone of the internet and communicates with each other. "Separation of duties" usually creates a well balanced system to maximize both efficiency and privacy.

Occam's Razor

Each blockchain has different usages and applications, and should be designed and optimized toward different directions. For example, a blockchain that is dedicated to relaying transactions between its subchains does not need to have Turing-complete contract running on top of it; another blockchain that connects devices in the same trust zone should not care about transactional privacy too much.

IoT Friendly

As aforementioned, IoT world is full of heterogeneous systems and nodes, stronger or weaker in terms of their resources of computation, storage and power. Since opera-

tions that can be done by weak nodes can be easily done by strong nodes, operations on the chain should be designed and optimized for weak nodes, *i.e.*, operations should be lightweight enough to conserve resources like computation, storage and energy.

4.2 Architecture: Blockchains in Blockchain

IoTeX is a network of many blockchains that are hierarchically arranged, where many blockchains can run concurrently with one another while retaining interoperability. In the IoTeX world, as shown in Figure 1, the *root blockchain* manages many independent blockchains, or *subchains*. A subchain connects to and interacts with IoT devices that share something in common, *e.g.*, they have a similar functional purpose, operate in similar environments, or share the similar level of trust. If a subchain does not function well, *e.g.*, being attacked or experiencing software bugs, the rootchain is completely unaffected. In addition, cross blockchain transactions are supported to transfer value and data from subchains to the rootchain or from one subchain to another via the rootchain.

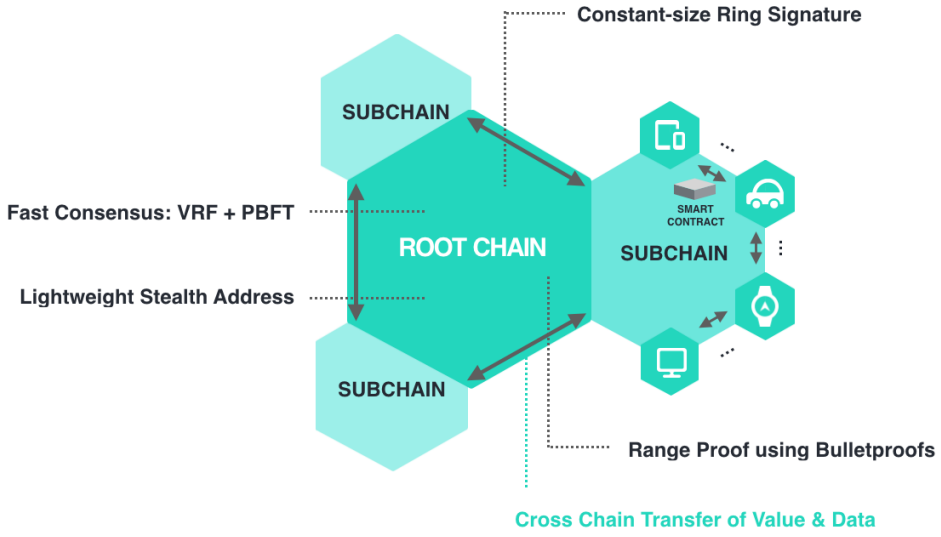

Figure 1: IoTeX: Blockchains in blockchain, a rootchain and subchains architecture.

The root blockchain is a public chain accessible by anyone, which has three main objectives:

1. **Relay** value and data across subchains in a privacy-preserving way to enable interoperability among subchains;

Table 2: Comparison Between Rootchain and Subchain

Properties	**Rootchain**	**Subchain**
Public vs Private	Public	Public or Private
Scalable	Required	Varies
Robust	Strongly Required	Required
Privacy-centric	Required	Varies
Extensibility	Non-Turing Complete	Turing Complete
Instant Block Finality	Required	Required

2. **Supervision** of subchains, *e.g.*, penalize the bonded operators of subchain by bond confiscation;

3. **Settlement and anchoring** of payments and trust for subchains.

With these defined objectives, the rootchain specifically focuses on scalability, robustness, privacy-preserving functions and the ability to orchestrate subchains.

A subchain, on the other hand, could potentially be a private blockchain and relies on the rootchain as relay to interact with other subchains. A subchain desires flexibility and extensibility to adapt diversified requirements of different IoT applications. A subchain is very likely run by operators whose role is contingent upon a sufficiently high bond being deposited on the rootchain. Optionally, the system allows operators to nominate one or more operators to act for it with or without extra bond. The operator acts like a light client on the rootchain, and a full node on the subchain to seal new blocks.

In all, the properties of the rootchain and subchains are summarized in Table 2.

4.3 Root Blockchain

The root blockchain uses UXTO-based model as Bitcoin [21] and Monero [20] for the following reasons:

- Transaction ordering becomes trivial without the need for nonce or sequence numbers, which places minimal demands on consensus schemes and allows transactions to be processed in parallel;

- Applying existent privacy-preserving techniques such as ring signature, and ZK-SNARKs for hiding sender, receiver and transaction amount becomes possible.

The root blockchain is composed of hash-linked blocks, and a block is composed of a header that hash-links to the previous block and a list of transactions. The

rootchain allows primarily two types of transaction: (1) basic transactions including `P2PKH`, `P2SH`, `Multisig` and *etc.*, and advanced transactions that enable cross blockchain operations like `BondedRegistration`, `Lock`, `ReLock`, `Reorg` and *etc.*. Validated transactions are added into a block that has a dynamic size, upper-bounded by 8MB. A block is produced every three seconds by our consensus scheme as detailed in the next section. The rootchain is designed to be non-Turing-complete with the support of a stack-based script and rich set of opcodes.

4.4 Subchains

IoTeX comes with a framework for developing and provisioning a tailored subchain for decentralized IoT applications by encapsulating low layer primitives like gossip protocol and consensus mechanism. The permission model, specification, parameters, and transaction types of the subchain can be customized to fit into its application.

IoTeX subchains use account-based model, which is better for tracking state transitions. There are two types of accounts, similar with Ethereum, regular accounts and contracts. Valid transactions are added into the block, which is produced by the same consensus scheme as the root chain to achieve the same degree of finality to make cross-blockchain communication more efficient. Subchains either use the root chain's token, IoTeXtoken, or define their own token. The token defined by developers on subchains can be distributed publicly by token sales or exchanging on public traded exchanges.

Smart contract is supported by subchains and runs on top of the lightweight and efficient virtual machine. We are currently evaluating Web Assembly (WASM) [36], an emerging web standard for building high-performance web applications. WASM is efficient and fast, and can be made deterministic and sandboxed with some modifications as attempted by EOS project [9]. Other options are also being explored. With smart contract, IoT devices connected to the same subchain utilize the shared state in two ways,

- First, devices can interact with the physical environment based on their subchains' states, *e.g.*, light bulbs turn on and off by themselves based on a "clock state" on the subchain;
- On the other hand, devices can mutate the state on subchains when the physical environment changes, *e.g.*, thermostat updates temperature via smart contract based on its sensor data.

4.5 Cross-Blockchain Communication

Cross-blockchain communication is expected to be used with high frequency in IoT applications. There is always the need for an IoT device in a subchain to coordinate with another device in a different subchain. Again, limited by IoT devices' low computation and storage footprint, we are motivated to design cross-blockchain communication in a fast and cost-effective way.

Pegging and Block Finality

Pegging is a mechanism for scaling the Bitcoin network via sidechains, originally proposed in [1]. It heavily relies on Simplified Payment Verification (SPV) [21], and allows Bitcoins to effectively "move" from the Bitcoin blockchain to the sidechain and back. The underlying idea is simple: Tokens are sent to a special address to be locked up on the Bitcoin blockchain; once this `Lock` transaction has been confirmed, one sends `Reorg` transaction to the sidechain including the `Lock` transaction and a proof of inclusion in the form of a Merkle branch. The sidechain uses SPV to verify `Reorg` transaction and, if it is validated, creates the same amount of tokens and sends them to a desired address on the sidechain. As of today, pegging serves as a primitive for almost all cross-blockchain communication protocols, *e.g.*, Cosmos, Lisk, Rootstock. Two separate pegging flows can be easily coupled together to make the so-called Two-Way Pegging (2WP) to make transfer tokens back and forth.

Block finality is the guarantee that the new block generated is final and cannot be changed. Block finality impacts the concrete implementation of pegging substantially as one has to wait until block finality is achieved (at least with high pliability) on the sending blockchain before requesting to `Reorg`. Most public blockchains like Bitcoin do not have instant finality. The receiving blockchain can only get a probabilistic assurance, *e.g.*, as more PoW miners confirm a transaction, it is more probable the transaction has been accepted. Utilizing a finalizing consensus addresses this problem because the receiving chain has assurance with one block confirmation on the sending blockchain. For IoT applications, cross-blockchain transferring of value and data is expected to be fast and cost-effective, which requires a finalizing consensus mechanism on both rootchain and subchains. IoTeX consensus achieves instant block finality, detailed in the next section.

Cross-Blockchain Communication Protocol

Let's review the protocol at a high level by assuming an address named Charlie on subchain 1 wishes to dispatch a transaction to an address named David on subchain 2, and all three blockchains use the same type of token without transaction fee for simplicity. Note that by applying pegging naively, four transactions are needed to

make a "remote call" from subchain 1 to subchain 2 via rootchain, *i.e.*, (1) a `Lock` transaction on subchain 1; (2) a `Reorg` transaction against rootchain; (3) another `Lock` transaction on rootchain; and (4) another `Reorg` transaction against subchain 2. This process indicates David has to wait for at least 4 blocks to accept this "remote call", and data this "remote call" carries needs to be stored on all three blockchains, which makes it slow and expensive. We aim to optimize this process by combing (2) and (3) into one `ReLock` transaction, which not only speeds up the entire process but also avoids storing data on subchain 1 and the rootchain. Our protocol is depicted in Figure 2.

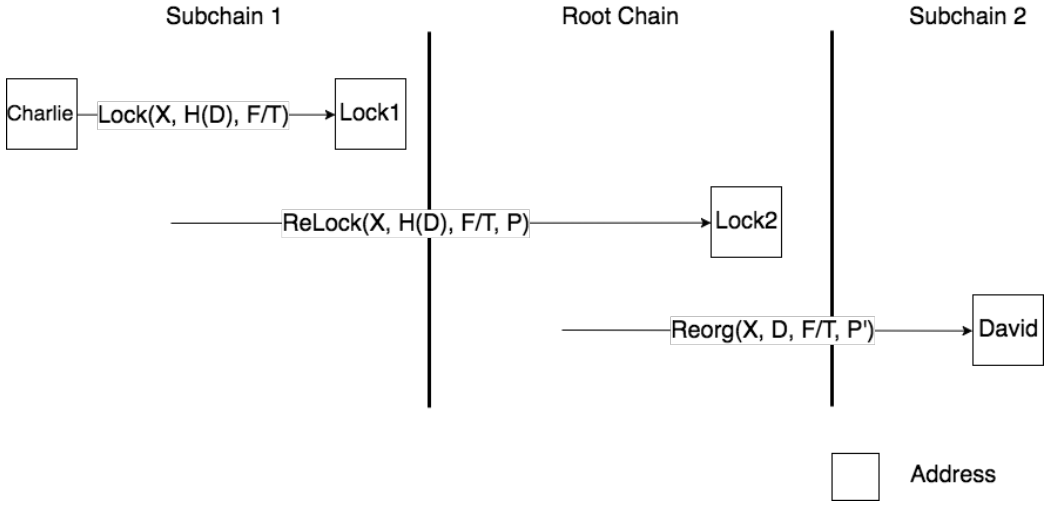

Figure 2: Cross-Blockchain Transactions

IoTeX cross-blockchain protocol has the following steps.

1. Each subchain is registered on the rootchain by submitting a transaction called `BondedRegistration` to the rootchain, including its chain name, chain ID, configuration, genesis block, and nomination of operators; This is a one-time process;

2. When Charlie wishes to dispatch a transaction to David, he initiates a `Lock(`X`,` $H(D)$`,` F/T`)` transaction where X is the amount of tokens, $H(D)$ is the hash of the data D to be attached, F/T indicates the from and to addresses including IDs for both chains;

3. Once `Lock` transaction has been included on subchain 1, Charlie initiates `ReLock(`X`,` $H(D)$`,` F/T`,` S`,` P`)` transaction to the rootchain by including X, $H(D)$, F/T, some current stats of subchain 1 denoted as S as well as proof-of-inclusion P

that includes Merkle branches of recent block headers and Merkle branches proving `Lock` transaction has been included;

4. The rootchain validates `ReLock` transaction and accepts it by including it in the latest block, and creates X tokens and locked them in a special address;

5. Once `ReLock` transaction has been included on the rootchain, Charlie broadcasts a `Reorg(`X`,` D`,` F/T`,` P'`)` transaction on rootchain's network with X, D, F/T and another proof-of-inclusion P' that proves the inclusion of `ReLock` transaction;

6. Operators of subchain 2 become aware of `Reorg` transaction, and they validate and create the same amount of tokens on subchain 2 and send them to address David with D associated.

Sharing the Root Blockchain's "Bandwidth"

One possible concern regarding cross-blockchain communication is that a malicious subchain spams the rootchain or another subchain by sending over a huge amount of cross-blockchain transactions that exhausted other blockchains' capacity. One way to mitigate is to let each subchain bid for its quota and "rate-limit" transactions from a subchain if its quota is exhausted.

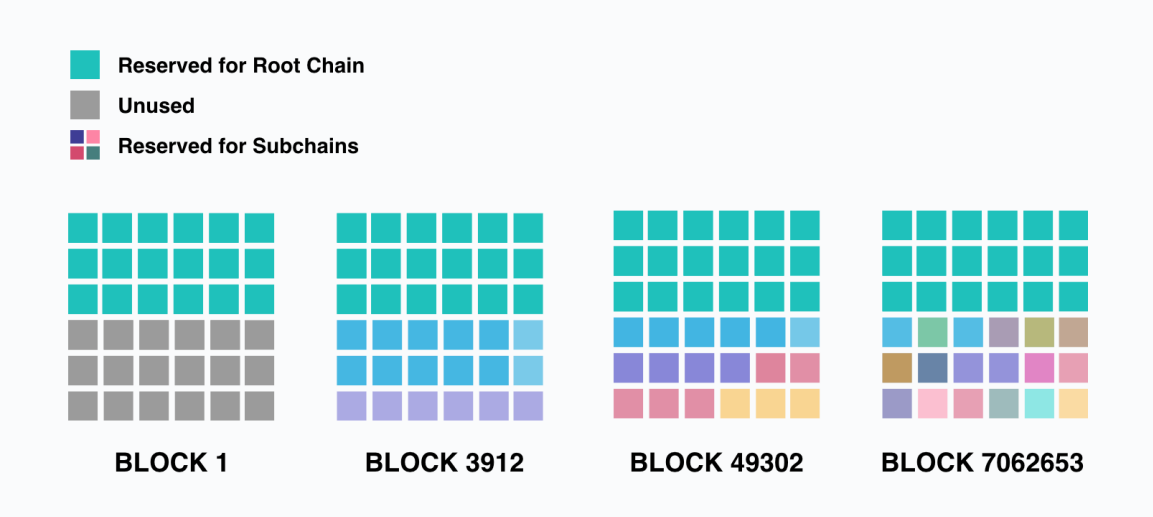

Figure 3: Bandwidth Model for Sharing Rootchain Capacity

One can define quota based on the storage space within one block. Assuming block size is 8MB maximum, and 4MB is reserved for normal transactions happening on the

Table 3: Privacy-Preserving Techniques for Blockchains

Techinique	Hide Sender	Hide Receiver	Hide Amount
Stealth Address	N	Y	N
Pedersen Commitments	N	N	Y
Ring Signatures	Y	N	N
zk-SNARKs	Y	N	Y

rootchain, and 4MB is reserved for all cross-blockchain transactions, which is further divided into, say 4096 quota piece with each quota piece to be 1KB. A subchain bids for n quota pieces (with a certain upper bound) according to the intended usage by putting down a deposit proportional to n. At each round, only up to nKB can be used within a new block for transactions from this subchain and each such transaction is charged a "bandwidth" fee from the deposit (to reward miners who help to enforce this rule); remaining transactions are queued up and eventually dropped when timeout. The quota allocation could be dynamic in the sense that it gets changes when the rootchain grows, as shown in Figure 3. If one subchain spams others, it burns out its deposit at a fast pace and eventually loses the quota. On the other hand, if one subchain puts down a big deposit merely to reserve a big chunk of bandwidth without actually using it, the rootchain will have a mechanism to refund part of the deposit based on the ratio between the average number of transactions per block and the reserved bandwidth, which helps to stabilize the reserved bandwidth close to the actual usage.

5 Built-in Privacy-Preserving Transaction

The privacy provided natively by Bitcoin and Ethereum is limited to pseudonymity. Transaction details are not confidential. The transaction amount and the assets being transferred, its metadata, and its relationships to other transactions, can be trivially learned by anyone. In fact, there are three aspects of privacy, sender privacy, receiver privacy and privacy of transaction details in this context. Various cryptographic schemes can be applied to address them, as shown in Table 3.

IoTeX integrates stealth address for receiver privacy, ring signature for sender privacy and Pedersen Commitments for hiding transaction amount with the following innovations and improvements:

- A lightweight stealth address scheme is designed to exempt receivers from scanning the entire blockchain to be aware of incoming transactions;

- Ring signature is optimized to make it compact in size with a distributed trusted setup.

5.1 Hide Transaction Receiver with Relayable Payment Code

Stealth Address

Stealth address technique originated from Cryptonote protocol [28], which solves the receiver problem using a "half round" Diffie–Hellman key exchange protocol. Assuming Bob wants to hide the fact that he receives tokens from Alice, here is how it works:

1. Bob creates two pairs of private and public keys, denote as (a, A) and (b, B), where $A = a \cdot G$ and $B = b \cdot G$, where G is the base point on an elliptic curve.

2. Bob publishes public keys (A, B) which are known as his stealth address;

3. Alice calculates and sends tokens to $P = H(rA) \cdot G + B$ using a hash function H, a random big number r and Bob's stealth address B. This transaction is broadcast along with $R = r \cdot G$;

4. Bob watches all transactions, calculates $P' = (H(aR) + b) \cdot G$ (since he knows a, b, R and G) with the hope that P' equals to P. If $P' = P$, Bob could spend tokens send to P' with private key $H(aR) + b$.

One obvious drawback of stealth address it that the receiver has either to scan all transactions (which is not ideal in an IoT world) in the network or rely on the assistance of a trusted full node (which compromises privacy to a certain degree).

Payment Code

Payment code has been designed to address the above drawback of stealth address with a certain sacrifice in privacy. The idea is that Alice notifies Bob of a payment code in a confidential way and Bob only watches transactions against addresses deriving from that code. Therefore, this proposal has two flows – notification, which is a one-time setup between two certain parties, and sending, which can happen multiple times between these two parties.

Assuming Alice has master public-private key pair $(mpub_{Alice}, mpri_{Alice})$ where $mpub_{Alice} = mpri_{Alice} \cdot G$ and wallet public-private key pair $(wpub_{Alice}, wpri_{Alice})$ where $wpub_{Alice} = wpri_{Alice} \cdot G$; Bob has master public-private key pair $(mpub_{Bob}, mpri_{Bob})$ where $mpub_{Bob} = mpri_{Bob} \cdot G$, the one-time notification flow works as below:

1. Bob derives $B_0 = b_0 \cdot G = (mpri_{Bob} + Hash(0, seed, metadata)) \cdot G$, converts it to an notification address $\texttt{addr}(B_0)$, publishes it and listens on it

2. Alice picks a chain code cc at random; $(mpub_{Alice}||cc)$ is the payment code for Alice;

3. Alice calculates a shared secret $S = wpri_{Alice} \cdot B_0$ and sends masked payment code $P' = (mpub_{Alice}||cc) \oplus HMAC512(xofS)$ to $\texttt{addr}(B_0)$;

4. Upon receipt, Bob learns $wpub_{Alice}$, and recovers $S = wpub_{Alice} \cdot b_0$, unmasks P' to obtain $(mpub_{Alice}||cc)$.

Once the notification flow is done, Alice and Bob establish one uni-directional private channel for sending tokens. The first sending flow works as below:

1. Alice derives a new address from the her payment code (that is already shared with Bob) by $A_0 = a_0 \cdot G = mpub_{Alice} + Hash(0, seed, metadata) \cdot G$;

2. Alice selects the next unused public key derived from B_0. Note that B_0 is the unused public key for the first round.

3. Alice calculates the new shared secret $S_0 = a_0 \cdot B_0$, and calculates the ephemeral public key used to send the transaction to which is $B'_0 = B_0 + \texttt{SHA256}(S_0) \cdot G$

4. Bob could derive A_0 non-interactively since he knows Alice's payment code, and only listens on address derived from $B'_0 = B_0 + \texttt{SHA256}(S_0) \cdot G$ and $S_0 = A_0 \cdot b_0$.

5. Upon receipt, Bob could use the tokens with private key $b_0 + \texttt{SHA256}(S_0)$.

The following sending flows works similarly.

Bob does not need to scan or rely on a full node to scan all transactions. The notification transaction does leak the intention of Alice wants to send something to Bob, but the actual "sending of something" is hidden from everyone else.

Relayable Payment Code

To further minimize the privacy leak, we designed the relayable payment code on top of the original payment code proposal. While the sending flow remains the same, we improved the notification flow to make it possible for Alice to secretly share her payment code with Charlie without using the notification transaction, assuming Alice and Bob have one uni-directional private channel, and Bob and Charlie have another uni-directional private channel. To achieve that, we leverage Hashed Timelock Contracts (HTLCs), which require that the receiver of a payment either acknowledge receiving

the payment prior to a deadline by generating cryptographic proof of payment or forfeit the ability to claim the payment, returning it to the sender.

Assuming Charlie has master public-private key pair $(mpub_{Charlie}, mpri_{Charlie})$ where $mpub_{Charlie} = mpri_{Charlie} \cdot G$. The improved notification flow works as follows.

1. Charlie derives $C_0 = c_0 \cdot G = (mpri_{Charlie} + Hash(0, seed, metadata)) \cdot G$, converts it to an notification address $\texttt{addr}(C_0)$, publishes it. Note that C_0 is published for Alice to calculate the shared secret, but not for receiving any transactions;

2. Alice generates her payment code $(mpub_{Alice}||cc)$ in the same way;

3. Alice calculates a shared secret $S = wpri_{Alice} \cdot C_0$ and sends masked payment code $P' = (mpub_{Alice}||cc) \oplus HMAC512(xofS)$ with X tokens as incentive and $HTLC(Hash^2(cc))$ to Bob using their uni-directional private channel, where $HTLC$, as part of the locking or redeem script, states that the tokens become spendable if the pre-image of $Hash^2(v)$ is given, i.e., $Hash(cc)$;

4. Bob, incentivized by the tokens sent over from Alice, sends P', $Y, Y < X$ tokens and $HTLC(Hash^2(v))$ to Charlie using their uni-directional private channel;

5. Charlie, upon receiving Bob's transaction, calculates $S = wpub_{Alice} \cdot c_0$ to unmask Alice's payment code, and spent the transaction by disclosing $Hash(cc)$, which makes Alice-to-Bob transaction spendable to reward Bob.

Once this flow is done, Alice and Charlie establish one uni-directional private channel for sending tokens. It is noteworthy that the routing of Alice's transaction could be multiple hops.

Our relayable payment codes offer better privacy in terms of hiding the intention of "sending of something" on the chain by leveraging the existing private channels without adding any computation or storage overhead to the nodes, which, while designed for IoT scenarios, is usable for most blockchains like Bitcoin.

5.2 Enable Confidential Transactions

5.2.1 Problem Statement

A typical transaction on the Bitcoin blockchain is shown in Figure 4. Essentially, a blockchain transactions is just a tuple $(\{pk_{in,i}\}, \{pk_{out,j}\}, \{v_{i,j}\})$, where $\{pk_{in,i}\}$ are input addresses, $\{pk_{out,j}\}$ are output addresses, and $\{v_{i,j}\}$ are transaction amounts among input and output addresses. Because Bitcoin transactions are stored in cleartext in the public ledger, it has raised a lot of security and privacy concerns.

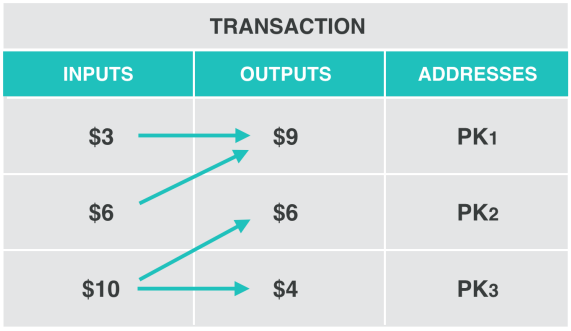

Figure 4: A Transaction on the Bitcoin Blockchain

The goal of confidential transactions (see Figure 5) is to enable *only* senders and receivers of transactions to reveal the $\{v_{i,j}\}$ values and conceal them from the rest of the world. Moreover, confidential transactions also allow other network entities to verify the validity of those transactions in question without seeing the actual amounts. The realization of confidential transactions on blockchain requires a number of advanced cryptographic techniques.

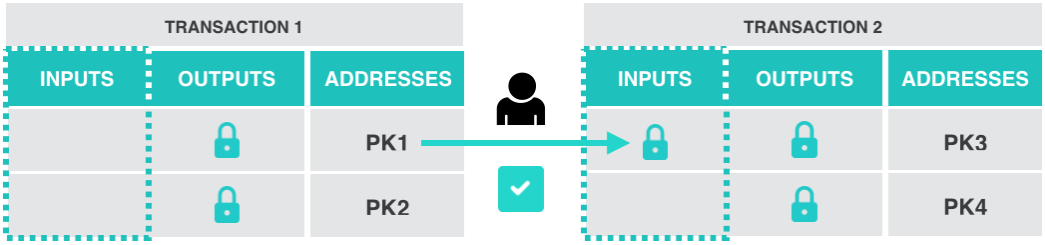

Figure 5: A Confidential Transaction with Public Verifiability

5.2.2 Cryptographic Building Blocks

Proof of Knowledge

A proof of knowledge, denoted by (P, V), is an interactive proof between a prover P and a verifier V, in which the prover wants to demonstrate that he knows some information. More specifically, P has (x, w) belonging to a relation R, where x is the problem and w is the solution (also called a *witness*). V knows x and he accepts only if P could convince V that he knows w.

Zero-Knowledge Proof

In a zero-knowledge proof protocol, the prover proves a statement to the verifier without revealing anything about the statement other than that it is true, which protects the prover against the malicious verifier, which attempts to gain more knowledge than what is intended. The protocol can be either *interactive* or *non-interactive*. The key difference with non-interactive proofs is that all interactions consist of a single message sent by the prover to the verifier. We use the notation $\mathsf{NIZKPoK}(\alpha, \beta) : a = g^{\alpha} \wedge b = g^{\beta}$ to denote a non-interactive, zero-knowledge proof of knowledge of the values α and β such that $a = g^{\alpha}$ and $b = g^{\beta}$. All values not enclosed in parenthesis are assumed to be known to the verifier. When we use a non-interactive zero-knowledge proof to authenticate auxiliary data, the resulting scheme is referred to as *signature of knowledge* [8]. Basically, a signature of knowledge scheme means that one in possession of a solution w to the problem x has singed the message m. For the above $\mathsf{NIZKPoK}$, we use notation $\mathsf{SoK}[m](\alpha, \beta) : a = g^{\alpha} \wedge b = g^{\beta}$ to denote a signature of knowledge on message m.

Ring Signature

The concept of ring signature was first introduced by Rivest *et al.*[27] in 2001 as a special kind of group signature. In a ring signature, the message signer selects a set of ring members including themselves as the possible message signers. The verifier can be convinced that the signature was indeed generated by one of the ring members. However, the verifier is not able to tell which member actually generated the signature. Unlike a general group signature, a ring signature scheme does not involve designating a group manager for managing the set of ring members, thereby eliminating the possibility of revealing the identity of the actual message signer by the the group manager. In order to provide anonymity in smart contract token transactions, a special kind of ring signature, so-called linkable ring signature, has been employed in the privacy-focused cryptocurrency Monero [20]. Linkable ring signature has additional property that any signatures generated by the same signer, whether signing the same message or disparate messages, has an identifier (called a tag) linking the signatures. This property enables third parties to efficiently verify that the signatures were generated by the same signer, without leaking the actual signer's identity. The linkable ring signature used in Monero is called a Multi-layered Linkable Spontaneous Anonymous Group Signature (MLSAG) [22], which is a ring signature on a set of key-vectors and has a communication complexity of $O(m(n+1))$, where m is the number of public/private key pairs owned by the signer and n is the size of ring.

Accumulator

One-way accumulators, which were first proposed by Benaloh and de Mare in [2], are defined as one-way hash functions with the property of being *quasi-commutative*. A quasi-commutative function $f : X \times Y \to X$ satisfies that, for all $x \in X$ and for all $y_1, y_2 \in Y$, we have $f(f(x, y_1), y_2) = f(f(x, y_2), y_1)$. A one-way accumulator allows us to combine a set of values into a secure digest and this digest does not depend on the order in which the values are accumulated . It can also be used to generate a witness that enables one to attest that a given value is actually part of the accumulator.

Commitment Scheme

A commitment scheme is a protocol enabling a user to commit to a value of his choice without revealing that value to the recipient of the commitment. In a later phase, when the user is asked to reveal the committed value, the recipient will have the means to verify that his revealed value is indeed unconditionally linked to his commitment. A commitment scheme should meet two requirements. While the *hiding* requirement prevents the recipient from learning the content of the commitment, the *biding* requirement prevents the user from cheating when opening this commitment. In Pedersen commitment scheme [23], the domain parameters are a cyclic group $\mathbb{G}$ of prime order q, and generators $(g_0, \ldots, g_m)$. For committing to the values $(v_1, \ldots, v_m) \in \mathbb{Z}_q^m$, one picks a random number $r \in \mathbb{Z}_q$ and set the commitment $C = \mathsf{PedCom}(v_1, \ldots, v_m; r) = g_0^r \prod_{i=1}^m g_i^{v_i}$.

5.2.3 Our Improvements

In [31], Sun *et al.*presented RingCT 2.0, which employed a cryptographic accumulator to further reduce the communication complexity to $O(n)$ at the cost of additional computations. We note that although RingCT 2.0 reduced the communication complexity significantly when compared to MLSAG, the domain parameter generation of the accumulator requires a one-time "trusted setup" process liked Zcash. Hence one has to trust that whoever generated the secret parameters destroy them when they are done, which has raised security and privacy concerns for the system. To address this issue, our solution is to employ a secure multi-party computation (SMPC) protocol among a set of bootstrapping nodes of the blockchain to generate secret domain parameters in a secure and distributed manner. In addition, the following directions are currently being investigated to improve the RingCT-like protocols in terms of communication and computational overhead:

- A new linkable ring signature scheme with communication complexity less than $O(n)$
- A new approach for aggregating multiple linkable ring signatures
- A sigma protocol for trustless setup of secret domain parameters

We aim to propose a novel confidential transaction solution that is able to achieve a good trade-off between communication and computation cost.

5.3 Prove Transaction Amount Range with Bulletproofs

As a drop-in replacement of Pedersen commitments, bulletproofs [5], a new non-interactive zero-knowledge proof protocol with very short proofs and without a trusted setup, has been proposed recently, which reduces the size of a range proof from linear to sublinear and further reduces the transaction size without additional computation overhead. Since Bulletproofs fit IoTeX's design principle well, we are going to integrate bulletproofs into IoTeX.

6 Fast Consensus with Instant Finality

6.1 Background

6.1.1 Proof of Work

Proof of Work (PoW) is the backbone to reach the global consensus of most blockchains, including Bitcoin and Ethereum. PoW makes it computationally difficult to construct a valid block and attach it to a blockchain. The longer the blockchain becomes, the harder it is to reverse any transaction recorded previously by the blockchain. To manipulate the blockchain, an attacker needs to own 51 percent of the whole computation power of a PoW-based blockchain network.

Although PoW provides an elegant solution for the global consensus of large distributed blockchains, it has several inherent drawbacks. The overall computation cost to maintain the global consensus is the same cost of the 51 percent attack. This means that even if the majority of the blockchain participants are honest, they still have to use a lot of electricity to maintain the blockchain, which is not suitable for the environment of IoT networks that usually favor energy efficiency. In addition, on the level of individual devices, computing PoW usually costs a lot of CPU cycles and memory usage, which poses difficult requirements to the hardware manufacturing and costs of embedded IoT devices. Last but not lease, PoW does not provide

instant finality which is a critical property required to construct efficient cross-chain communication.

6.1.2 Proof of Stake

Proof of Stake (PoS) was proposed as an efficient alternative to PoW for blockchains reaching consensus, which aims to avoid the above mentioned issues of PoW. The basic idea of PoS is that a randomly chosen set of nodes vote on the next block, and their votes are weighted based on the size their of deposits (i.e. stake). If certain nodes misbehave, they may lose their deposits. In this way, without computationally intensive PoW, the blockchain can run much more efficiently, and can achieve an economic stability: The more stake a participant has, the more incentive the node has to maintain the global consensus, and the less likely the node misbehaves. There are a couple of public PoS designs and implementations, such as Tendermint [32] that has been adopted by many applications [33].

6.1.3 Delegated Proof of Stake (DPoS)

Delegated Proof of Stake (DPoS) improves upon the idea of PoS in the way that DPoS allows participants to choose some delegates to represent their portions of stakes in the network. For example, Alice can send a message to the network to grant Bob the ability to represent her stake and vote on behalf of her. DPoS offers several benefits for our IoT applications:

- Small players can pool their stakes to have a higher chance together to participate in block proposing and voting, and share the rewards afterwards.
- Resource-constrained nodes can choose their delegates, so not all the nodes need to stay online to contribute to consensus.
- Delegates can be the nodes with strong power supply and network conditions, and also can be chosen dynamically and randomly, so we will have a higher overall availability for the network reaching consensus.

The typical cryptocurrencies using DPoS include EOS [9] and Lisk [18].

6.1.4 Practical Byzantine Fault Tolerance

Practical Byzantine Fault Tolerance (PBFT) was proposed by Castro and Liskov [7] in 1999 as an efficient and attack-resistant algorithm for reaching agreements in a distributed asynchronous network. We plan to use PBFT for the underlying voting algorithm of our DPoS consensus mechanism, because it is a concise and well-studied

algorithm that provides quick finality that is critically important for building an efficient and salable blockchain. As demonstrated in Castro and Liskov's original paper, PBFT offers both availability and safety if at most a third of the network nodes are faulty or malicious, and the network cost of PBFT is very minimum, i.e. about 3 percent compared to unreplicated network system.

The typical cryptocurrencies based on PBFT include Stellar [30] and Zilliaq [38].

6.2 Randomized Delegated Proof of Stake (Roll-DPOS)

To have a fast and efficient consensus mechanism with instant block finality in the context of IoT, we combine the concepts of DPoS, PBFT and Verifiable Random Functions (VRFs). VRF was first introduced by Micali *et al.* in [19] and is a family of functions that can produce publicly verifiable proofs for the correctness of their random outputs. At a high level, our proposed Roll-DPOS has four phases *elect candidates*, *form committee*, *propose block* and *finalize block*.

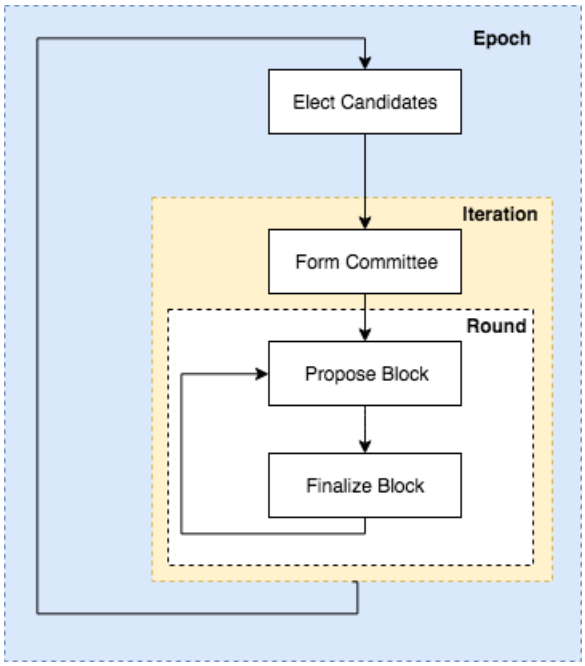

Figure 6: Randomized Delegated Proof of Stake (Roll-DPOS)

6.2.1 Elect Candidates

All notes in the IoTeX network could participate this phase in terms of voting for the committee candidates. To encourage nodes to vote, the system makes sure the

delegates share forged rewards with their voters. The candidates forms a set of at least 97 delegates; this number will increase in the future to further avoid the centralization of the mining power. Once the candidates are selected, they will be fixed in one epoch, which is consistent of 47 iterations.

6.2.2 Form Committee

In each iteration, a random committee of size 11 is selected from the candidate pool using VRF for creating blocks in the next 11 rounds. The idea is to use the hash of the block in the last iteration and the node's private key as inputs to the VRF to produce a Boolean output indicating if one is selected as the committee member, a priority indicating one's order to propose block and a proof indicating one's qualification for proposing the block at a certain round. The use of VRF is important as it provides an non-interactive way to sort all delegates for proposing blocks in a fairness and security way. To this end, we use the efficient VRF as being used in Algorand [12].

6.2.3 Propose Block

In each round (which is roughly every 3 seconds), every committee node proposes a new block and broadcasts it to the network, together with the priority and the proof. Only the block proposed by a committee node with highest priority and has not been proposed in the same iteration is considered by other nodes, which is called a candidate block.

6.2.4 Finalize Block

In the same round, all other nodes uses PBFT to vote for/against the candidate block. If more than 2/3 committee nodes agree candidate block's validness, it is finalized and is appended to the blockchain by everyone in the network. After that, *propose block* and *finalize block* are executed in the next round; if the current iteration finishes, another random committee will be formed before *propose block* and *finalize block* are executed.

6.3 Creating Periodic Checkpoints for Light Clients

In IoT networks, we expect a lot of devices are *light clients*, which are the blockchain participants that don't record the full transaction history locally. Considering the storage overhead of the full blockchain, e.g., over 100GB for Bitcoin [4], many embedded low-cost IoT devices may not have the capacity to download the full blockchain. However, these light clients still have ability to quickly verify the correctness of the

blockchain and interact with it - the design is included in Satoshi's original Bitcoin whitepaper [21].

However, using PoS instead of PoW has a disadvantage for light clients. When verifying correctness of PoS based blockchains, clients need to download a list of public keys and signatures for block proposers and voters, and the sets of block proposers and voters may change for each block. Thus, when light clients come back online after staying offline for a while, the clients may need to download a large number of public keys and signatures, and then verify all of them. To mitigate this performance issue, Vitalik, the inventor of Ethereum, has proposed creating periodic checkpoints on the blockchain, called *epochs* [6], for example every 50 blocks. Each checkpoint can be verified based on the previous checkpoint, such that light clients can catch up with the whole blockchain much faster.

7 Token on IoTeX Network

The native digital cryptographically-secured token of the IoTeX Network (IOTX) is a major component of the ecosystem on the IoTeX Network, and is designed to be used solely on the network. Prior to the launch of IoTeX mainnet, the token will exist as an ERC20 compatible token on the Ethereum blockchain, which will be migrated to a token on the IoTeX mainnet when the same is launched.

IOTX is required as virtual crypto "fuel" for using certain designed functions on the IoTeX Network (such as executing transactions and running the distributed applications on the IoTeX Network), providing the economic incentives which will be consumed to encourage participants to contribute and maintain the ecosystem on the IoTeX Network. Computational resources are required for running various applications and executing transactions on the IoTeX Network, as well as the validation and verification of additional blocks / information on the blockchain, thus providers of these services / resources would require economic incentives for the provision of these resources (i.e. "mining" on the IoTeX Network) to maintain network integrity, and IOTX will be used as the unit of exchange to quantify and pay the costs of the consumed computational resources. IOTX will be mineable for 50 years, with rewards of the mining reducing over time based on a linear gradient reduction model.

IOTX is an integral and indispensable part of the IoTeX Network, because in the absence of IOTX, there would be no common unit of exchange to pay for these costs, thus rendering the ecosystem on the IoTeX Network unsustainable.

IOTX is a non-refundable functional utility token which will be used as the unit of exchange between participants on the IoTeX Network. The goal of introducing IOTX is to provide a convenient and secure mode of payment and settlement between participants who interact within the ecosystem on the IoTeX Network. IOTX does

not in any way represent any shareholding, participation, right, title, or interest in IoTeX Foundation Ltd. (the **Foundation**), its affiliates, or any other company, enterprise or undertaking, nor will IOTX entitle token holders to any promise of fees, revenue, profits or investment returns, and are not intended to constitute securities in Singapore or any relevant jurisdiction. IOTX may only be utilized on the IoTeX Network, and ownership of IOTX carries no rights, express or implied, other than the right to use IOTX as a means to enable usage of and interaction with the IoTeX Network.

In particular, IOTX:

(a) is non-refundable and cannot be exchanged for cash (or its equivalent value in any other virtual currency) or any payment obligation by the Foundation or any affiliate;

(b) does not represent or confer on the token holder any right of any form with respect to the Foundation (or any of its affiliates) or its revenues or assets, including without limitation any right to receive future revenue, shares, ownership right or stake, share or security, any voting, distribution, redemption, liquidation, proprietary (including all forms of intellectual property), or other financial or legal rights or equivalent rights, or intellectual property rights or any other form of participation in or relating to the IoTeX Network, the Foundation, the Distributor and/or their service providers;

(c) is not intended to be a representation of money (including electronic money), security, commodity, bond, debt instrument or any other kind of financial instrument or investment;

(d) is not a loan to the Foundation or any of its affiliates, is not intended to represent a debt owed by the Foundation or any of its affiliates, and there is no expectation of profit; and

(e) does not provide the token holder with any ownership or other interest in the Foundation or any of its affiliates.

8 IoTeX Powered Ecosystems

The IoTeX blockchain supports a variety of IoT ecosystems, shared economies, smart home, autonomous vehicles, and supply chain, etc. Different types of developers leverage IoTeX in different ways. The developers supported by IoTeX include IoT hardware manufacturers, IoT device control system developers, smart home app developers, shared economies device manufacturers, supply chain data integrator, data

crowdsourcing vendors, autonomous cars developers, etc. This section describes a few IoTeX powered ecosystems.

8.1 Shared Economies

In recent years, many companies have focused on shared economies, from rides sharing such as Uber/Lyft/Didi, home sharing such as Airbnb, bikes sharing such as Mobike/ofo, to small item level sharing like battery bank, umbrella, etc. They all provide people with a better living, although some of them are suffering from their business models. It is a different topic to discuss their business models; here we mainly focus on their technological architecture. Among all shared economies, ride sharing is the one that can't avoid human operation, viz., drivers. It is not an IoT powered economy. However, in the future, when autonomous car technology becomes mature and popular, ride sharing will be powered by IoT.

All IoT powered shared economies share some similarities: They all require a lock that can be opened by a deposit and rental fee. It is very possible and also efficient to power the whole sharing and returning process using an IoT device. In centralized world, the economies are powered by a centralized cloud. There are various drawbacks:

1. A large deposit is held by a company that may not be trustworthy. Recently, there have been many cases where the company that runs a shared bike service in China can't pay back deposits to its users;

2. The shared economies are not completely driven by community. Many shared things are owned by a company. This has caused a waste of society resource. Take shared bikes as example. When the shared bikes companies are out of business, the bikes are disposed.

3. Due to the centralized nature, the user data will be stored and controlled by one company. There are risk that either the cloud or the client can be hacked to obtain user data.

IoTeX, as an infrastructure, could be utilized to power these applications without the issues above and make shared economies decentralized and more efficient. Concretely, an IoTeX-powered shared economy provides the following benefits:

1. Deposit is completely settled by smart contract. With no one holding back the money, returning of the deposit is always guaranteed. Users don't have to trust the company to use the service.

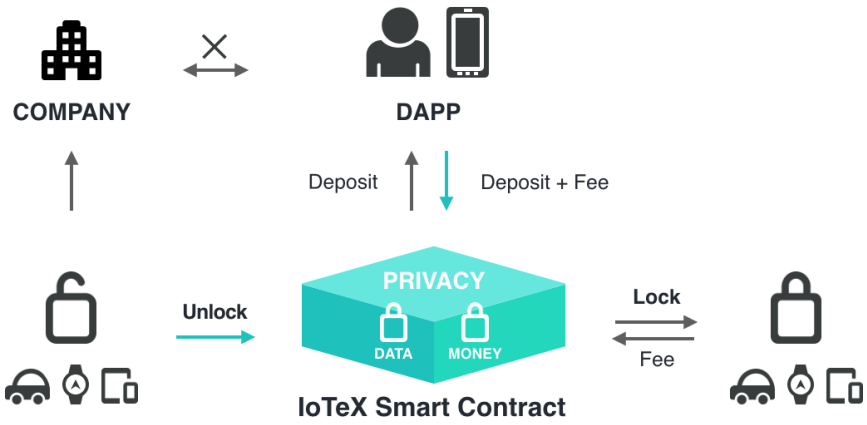

Figure 7: IoTeX Powered Shared Economy

2. Each shared thing realizes its value and mission in an autonomous way. In the ecosystem, it doesn't matter who owns the shared things in it. Everyone can own and contribute to the ecosystem. The economy can be run by community. As a result, companies can play a role of maintaining the IoT lock and manage the community. It is much lighter business model that companies can fast expand and serve more people.

3. Again, users don't have to trust the company to maintain their data. Their data is kept in the chain with privacy protection.

Fig. 8 describes how shared economy works based on IoTeX blockchain.

8.2 Smart Home

In the existing smart home market, many IoT device manufacturers are still using out-of-date technologies to develop their products. They need a large amount of development work on their cloud. The cost of development and maintenance is high, and performance is low because of the round trip required to the cloud. Deploying their products onto IoTeX blockchain will largely reduce operating cost on engineering and cloud computing, and at the same time, largely increase the performance of their devices. In a simple smart light bulb example, with cloud technology, it takes two

trips from user instruction to changing the state of a light bulb. Manufacturers are not cloud experts so often their service is not optimal. The round trip can take one to three seconds. This forces them to use cloud service by big IT companies. There are two downsides of using these cloud services:

1. Manufacturers can't fully control the availability of cloud services.
2. They need to continuously pay for the cloud service despite their one-time charge on selling their IoT devices.
3. There are risks of their cloud, client side, or intranet being hacked causing user data to be stolen or home security problems.

In contrast, IoTeX blockchain manages the devices locally and interact with public chain on the internet when necessary. The public chain is maintained by community. There is no maintenance cost for IoT manufacturers. IoTeX blockchain has privacy protection that can prevent leaking data or control being hacked even if the intranet is not safe.

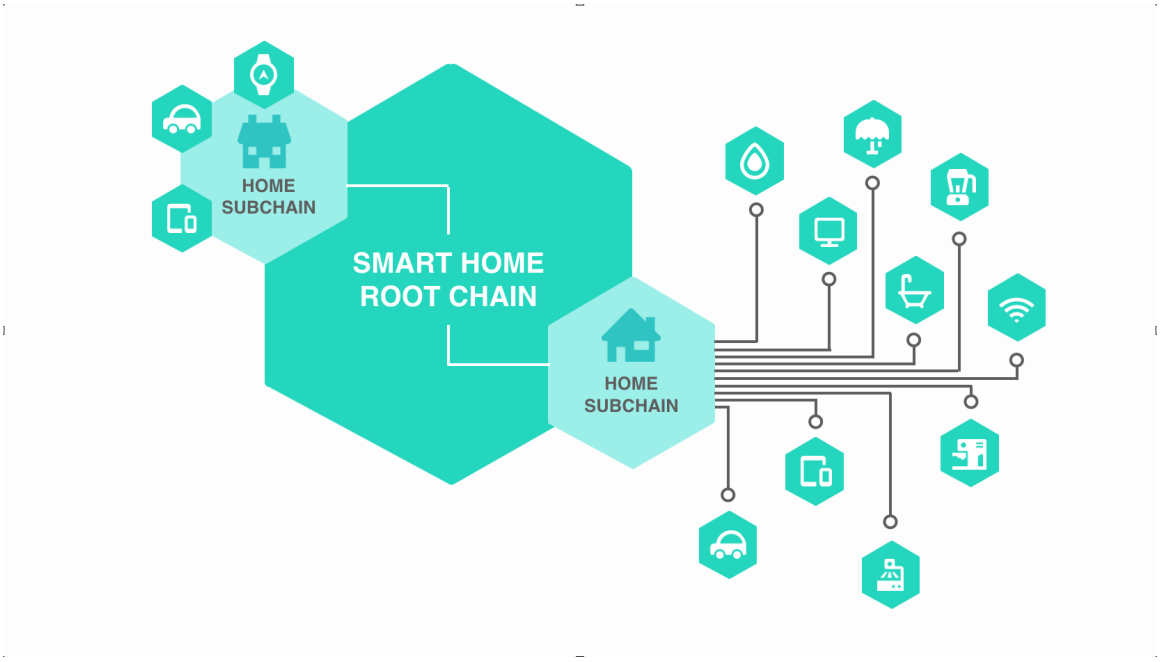

Figure 8: IoTeX-Powered Smart Home

In addition to allowing IoT manufacturers to deploy their IoT devices on IoTeX blockchain, IoTeX will partner with IoT chip makers to develop IoTeX blockchain-enabled chips to accelerate the design and manufacture cycles of IoT devices. IoT

manufacturers will simply integrate the chip to get their devices supported by IoTeX blockchain.

8.3 Identity Management

The growing world of IoT has impacted how Identity and Access Management (IAM) need to work. In terms of the identity of things, IAM must be able to manage user-to-device, device-to-device, and/or device-to-service/system. One straightforward way for identity management is to consider IoTeX blockchain as a decentralized PKI system (thanks to its immutability) where each entity is issued a cryptographic identity in the form of TLS certificate and the corresponding private. This certificate, which tends to be a short-lived one, is signed by the device's built-in and long-lived certificate and published on IoTeX blockchain (either rootchain or subchain). Peers and other entities can access and trust the short-lived certificate anchored on the blockchain, and things can then authenticate when they come online, ensuring secure communication between other devices, services and users, and prove their integrity.

In addition, the built-in and long-lived certificates for devices could be arranged in hierarchy, like the conventional PKI, where parent devices could sign children certificate. With the hierarchy, revoking and rotating of certificates becomes possible. For example, if one device gets compromised, its parent device or even if grandparent device could sign a revocation command and send to the blockchain where the latter invalidate the device's certificate.

9 Future Research Work

Some ongoing and future directions of research to improve IoTeX are as follows.

Privacy-preserving Computation

There are several areas in this direction we are actively exploring:

- How to retain confidential states on the blockchain which can be used for computing by a certain group of nodes;
- Privacy-preserving smart contract where smart contract can be evaluated when its business logic is protected by encryption. While fully homophobic encryption [26] and indistinguishable obfuscation schemes [11] are the holy grail in theory, practical proposals like Hawk [17] is promising for the near future;
- Further reduce the computation and storage footprint of the privacy-preserving techniques IoTeX is currently using;

- The quantum-safe version of privacy-preserving techniques IoTeX is currently using such as quantum-safe ring signature.

States Pruning and Transferring

We are evaluating different ways to safely prune the states stored on subchains to reduce the storage footprint since many IoT devices have limited storage. Compression of blocks and transactions is definitely a low-hanging fruit. Besides, transferring states from subchain to rootchain (since the latter is stronger in terms of storage) in a efficient and privacy-preserving way is also an interesting topic to investigate.

Governance and Self-amending

While IoTeX blockchains offer incentives for maintaining consensus on its ledgers, it does not have a on-chain mechanism for now that seamlessly amends the rules governing its protocol and rewards protocol development. We plan to conduct research on governance and self-amending to address this.

Tree Structured Blockchains

The current IoTeX is a two-layer blockchain and naturally, it should be extended to a tree of blockchains by leveraging techniques like Plasma and Cosmos. The plan is to evaluate these proposal and enhance the current design of IoTeX to eventually support more complex hierarchical structures.

10 Conclusion

In this white paper, we introduced IoTeX, a scalable, private, and extensible blockchain dedicated for Internet of Things, with its architecture and core technologies including 1. blockchains in blockchain to maximize scalability and privacy, 2. true privacy on blockchain based on relayable payment code, constant-size ring signature without trusted setup, and first implementation of bulletproofs, 3. fast consensus with instant finality based on VRF and PoS for high throughput and instant finality, and 4. flexible and lightweight IoTeX-based system architectures.

11 Acknowledgements

We would like to express our gratitude to our mentors and advisers and to the many people in the IoT, cryptography and cryptocurrency communities for their early feedback and constructive suggestions.

References

[1] Adam Back et al. "Enabling blockchain innovations with pegged sidechains". In: *URL: http://www. opensciencereview. com/papers/123/enablingblockchain-innovations-with-pegged-sidechains* (2014).

[2] Josh Benaloh and Michael de Mare. "One-Way Accumulators: A Decentralized Alternative to Digital Signatures". In: *Advances in Cryptology — EUROCRYPT '93: Workshop on the Theory and Application of Cryptographic Techniques Lofthus, Norway, May 23–27, 1993 Proceedings*. Ed. by Tor Helleseth. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 274–285. ISBN: 978-3-540-48285-7. DOI: 10.1007/3-540-48285-7_24. URL: https://doi.org/10.1007/3-540-48285-7_24.

[3] *Bitcoin Improvement Proposals*. https://github.com/bitcoin/bips.

[4] *Blockchain Size*. https://blockchain.info/charts/blocks-size.

[5] Benedikt Bünz et al. *Bulletproofs: Efficient Range Proofs for Confidential Transactions*. Cryptology ePrint Archive, Report 2017/1066. https://eprint.iacr.org/2017/1066. 2017.

[6] Vitalik Buterin. *Light Clients and Proof of Stake*. https://blog.ethereum.org/2015/01/10/light-clients-proof-stake/.

[7] Miguel Castro, Barbara Liskov, et al. "Practical Byzantine fault tolerance". In: *OSDI*. Vol. 99. 1999, pp. 173–186.

[8] Melissa Chase and Anna Lysyanskaya. "On Signatures of Knowledge". In: *Advances in Cryptology - CRYPTO 2006: 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006. Proceedings*. Ed. by Cynthia Dwork. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 78–96. ISBN: 978-3-540-37433-6. DOI: 10.1007/11818175_5. URL: https://doi.org/10.1007/11818175_5.

[9] *EOS*. https://eos.io/.

[10] AB Ericsson. "Ericsson mobility report: On the pulse of the Networked Society". In: *Ericsson, Sweden, Tech. Rep. EAB-14* 61078 (2015).

[11] Sanjam Garg et al. "Candidate indistinguishability obfuscation and functional encryption for all circuits". In: *SIAM Journal on Computing* 45.3 (2016), pp. 882–929.

[12] Yossi Gilad et al. "Algorand: Scaling byzantine agreements for cryptocurrencies". In: *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM. 2017, pp. 51–68.

[13] *HDAC Blockchain for IoT*. https://hdac.io/.

[14] *Hyperledger Fabric*. https://www.ibm.com/blockchain/hyperledger.html.

[15] *Internet of Things (IoT) Market by Software Solution (Real-Time Streaming Analytics, Security Solution, Data Management, Remote Monitoring, and Network Bandwidth Management), Service, Platform, Application Area, and Region - Global Forecast to 2022*. https://www.jasper.com/sites/default/files/cisco-jasper-hidden-costs-of-delivering-iiot-services-en_2.pdf. 2016.

[16] *ITC Blockchain for IoT*. https://iotchain.io/.

[17] Ahmed Kosba et al. "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts". In: *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE. 2016, pp. 839–858.

[18] *Lisk*. https://lisk.io/.

[19] Silvio Micali, Michael Rabin, and Salil Vadhan. "Verifiable random functions". In: *Foundations of Computer Science, 1999. 40th Annual Symposium on*. IEEE. 1999, pp. 120–130.

[20] *Monero – Private Digital Currency*. https://getmonero.org/.

[21] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. 2008.

[22] Shen Noether and Adam Mackenzie. "Ring Confidential Transactions". In: *Ledger* Vol. 1 (2016), pp. 1–18. DOI: https://doi.org/10.5195/ledger.2016.34.

[23] Torben Pryds Pedersen. "Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing". In: *Advances in Cryptology — CRYPTO '91: Proceedings*. Ed. by Joan Feigenbaum. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 129–140. ISBN: 978-3-540-46766-3. DOI: 10.1007/3-540-46766-1_9. URL: https://doi.org/10.1007/3-540-46766-1_9.

[24] Serguei Popov. "The tangle". In: *IOTA* (2016).

[25] *Raiden Network*. https://raiden.network/.

[26] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. "On data banks and privacy homomorphisms". In: *Foundations of secure computation* 4.11 (1978), pp. 169–180.

[27] Ronald Rivest, Adi Shamir, and Yael Tauman. "How to leak a secret". In: *Advances in Cryptology—ASIACRYPT 2001* (2001), pp. 552–565.

[28] Nicolas van Saberhagen. *Cryptonote v 2. 0*. 2013.

[29] Samsung. *Samsung ARTIK and Successful Strategies for Industrial IoT Deployment*. Samsung, 2016.

[30] *Stellar*. https://www.stellar.org/.

[31] Shi-Feng Sun et al. "RingCT 2.0: A Compact Accumulator-Based (Linkable Ring Signature) Protocol for Blockchain Cryptocurrency Monero". In: *Computer Security – ESORICS 2017: 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II*. Ed. by Simon N. Foley, Dieter Gollmann, and Einar Snekkenes. Cham: Springer International Publishing, 2017, pp. 456–474. ISBN: 978-3-319-66399-9. DOI: 10.1007/978-3-319-66399-9_25. URL: https://doi.org/10.1007/978-3-319-66399-9_25.

[32] *Tendermint*. https://tendermint.com/.

[33] *Tendermint Ecosystem*. https://tendermint.readthedocs.io/en/master/ecosystem.html.

[34] *Tezos: A new digital commonwealth*. https://www.tezos.com/.

[35] *The hidden costs of delivering IIoT services*. https://www.jasper.com/sites/default/files/cisco-jasper-hidden-costs-of-delivering-iiot-services-en_2.pdf. 2017.

[36] *WebAssembly*. http://webassembly.org/.

[37] Jan Henrik Ziegeldorf, Oscar Garcia Morchon, and Klaus Wehrle. "Privacy in the Internet of Things: threats and challenges". In: *Security and Communication Networks* 7.12 (2014), pp. 2728–2742.

[38] *Zilliqa*. https://www.zilliqa.com/.